Intel® Celeron® M Processor

Specification Update

October 2005

Notice: The Intel® Celeron® M processor may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are documented in this Specification Update.

Document Number: 300303-015

Contents

§

Revision History

Revision Number	Description	Date
-001	Initial Release	January 2004
-002	Revisions include: • Added errata WW22- W23 • Added Specification Clarification W1 • Updated Processor Identification table	April 2004
-003	Revisions include: • Added errata WW24- W25	May 2004
-004	Revisions include: • Updated Processor Identification table	June 2004
-005	Revisions include: • Change to Title to reflect processors (adding Celeron M processor ULV on 90 nm processor) • Processor Identification table to include Celeron processor ULV on 90 nm process • Added errata W26 and W27	July 2004
-006	Revisions include: • Updated General information (see Figure 3) for Celeron processor 350 and 360 • Processor Identification table to include Celeron processor 350 and 360	August 2004
-007	Revisions include: • Added errata W28 through W33	November 2004
-008	Revisions include: • Added errata W34 and W35	December 2004
-009	• Updated Processor Identification Table: Added C-0 S-Specs • Updated Summary Tables of Changes • Added Erratum W36 – W38	February 2005
-010	• Updated Processor Identification Table	March 2005

Revision Number	Description	Date

-011	• Updated Summary Tables of Changes
	• Updated Processor Identification Table:
	  o Added Celeron M processor ULV 383
	  o Updated Celeron M processor 360J & 350J
	• Added Errata W39
	• Added Specification Clarification W1
	• Added Specification Clarification W2
		May 2005

-012	• Updated Summary Tables of Changes • Removed Erratum W28 – W30 (which were duplicates of W3 – W5) • Added Erratum W40 – W42	June 2005
-013	• Updated Processor Identification Table: o Added Celeron M processor 380	July 2005
-014	• Updated Affected and Related Documents Tables • Updated Processor Identification Table 1	July 2005
-015	• Updated Processor Identification Table 1	October 2005

§

Preface

This document is an update to the specifications contained in the documents listed in the following Affected Documents/Related Documents table. It is a compilation of device and document errata and specification clarifications and changes, and is intended for hardware system manufacturers and for software developers of applications, operating system, and tools.

Information types defined in the Nomenclature section of this document are consolidated into this update document and are no longer published in other documents. This document may also contain information that has not been previously published.

Affected Documents

Document Title	Document Number
Intel® Celeron® M Processor Datasheet	300302-003
Intel® Celeron® M Processor on 90 nm Process Datasheet	303110-006

Related Documents

Document Title	Document Number
IA-32 Intel® Architecture Software Developer's Manual, Volume 1: Basic Architecture	253665
IA-32 Intel® Architecture Software Developer's Manual, Volume 2A: Instruction Set Reference, A-M	253666
IA-32 Intel® Architecture Software Developer's Manual, Volume 2B: Instruction Set Reference, N-Z	253667
IA-32 Intel® Architecture Software Developer's Manual, Volume 3: System Programming Guide	253668

Nomenclature

S-Spec Number is a five-digit code used to identify products. Products are differentiated by their unique characteristics, e.g., core speed, L2 cache size, package type, etc. as described in the processor identification information table. Care should be taken to read all notes associated with each S-Spec number.

Errata are design defects or errors. Errata may cause the Intel® Celeron® M processor's behavior to deviate from published specifications. Hardware and software, designed to be used with any given processor, must assume that all errata documented for that processor are present on all devices unless otherwise noted.

Documentation Changes include typos, errors, or omissions from the current published specifications. These changes will be incorporated in the next release of the specifications.

Specification Clarifications describe a specification in greater detail or further highlight a specification's impact to a complex design situation. These clarifications will be incorporated in the next release of the specifications.

Specification Changes are modifications to the current published specifications for the Celeron M processor. These changes will be incorporated in the next release of the specifications.

Summary Tables of Changes

The following table indicates the errata, documentation changes, specification clarifications, or specification changes that apply to the Celeron M processor. Intel intends to fix some of the errata in a future stepping of the component and to account for the other outstanding issues through documentation or specification changes as noted. This table uses the following notations:

Codes Used in Summary Table

Stepping

X:	Erratum, Specification Change or Clarification that applies to this stepping.
(No mark) or (Blank Box):	This erratum is fixed in listed stepping or specification change does not apply to listed stepping.

Status

Doc:	Document change or update that will be implemented.
PlanFix:	This erratum may be fixed in a future of the product.
Fixed:	This erratum has been previously fixed.
NoFix:	There are no plans to fix this erratum.

Row

Shaded:	This item is either new or modified from the previous version of the document.

Each specification update item is prefixed with a capital letter to distinguish the product. The key below details the letters that are used in Intel's microprocessor specification updates:

A = Intel® Pentium® II processor
B = Mobile Intel® Pentium® II processor
C = Intel® Celeron® processor
D = Intel® Pentium® II Xeon® processor
E = Intel® Pentium® III processor
G = Intel® Pentium® III Xeon® processor
F = Intel® Pentium® processor Extreme Edition and Intel® Pentium® D processor
H = Mobile Intel® Celeron® processor at 466 MHz, 433 MHz, 400 MHz, 366 MHz, 333 MHz, 300 MHz, and 266 MHz
J = 64-bit Intel® Xeon® processor MP with 1 MB L2 Cache
K = Mobile Intel® Pentium® III Processor – M
L = Intel® Celeron® D processor
M =Mobile Intel® Celeron® processor
N = Intel® Pentium® 4 processor

O = Intel® Xeon® processor MP
P = Intel® Xeon® processor
Q = Mobile Intel® Pentium® 4 processor supporting Hyper-Threading Technology† on 90-nm technology process
R = Intel® Pentium® 4 processor on 90 nm process
S = 64-bit Intel® Xeon® processor with 800 MHz system bus (1 MB and 2 MB L2 cache versions)
T = Mobile Intel® Pentium® 4 processor – M
U = 64-bit Intel® Xeon® processor MP with up to 8 MB L3 Cache
V = Mobile Intel® Celeron® processor on .13 Micron process in Micro-FCPGA Package
W = Intel® Celeron® M processor
X = Intel® Pentium® M processor on 90 nm process with 2-MB L2 Cache
Y = Intel® Pentium® M processor
Z = Mobile Intel® Pentium® 4 processor with 533 MHz System Bus

Note: The specification updates for the Pentium® processor, Pentium® Pro processor, and other Intel products do not use this convention.

NO.	Steppings			Plans	ERRATA
	B-1 CPU Signature= 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature= 0695h		
W1			X	NoFix	Performance Monitoring Event That Counts the Number of Instructions Decoded (D0h) Is Not Accurate
W2			X	NoFix	RDTSC Instruction May Report the Wrong Time-stamp Counter Value
W3			X	NoFix	Code Segment Limit Violation May Occur on 4-Gbyte Limit Check
W4			X	NoFix	FST Instruction with Numeric and Null Segment Exceptions May Cause General Protection Faults to Be Missed and FP Linear Address (FLA) Mismatch
W5	X	X	X	NoFix	Code Segment (CS) Is Wrong on SMM Handler When SMBASE Is Not Aligned
W6	X	X	X	NoFix	IFU/BSU Deadlock May Cause System Hang
W7			X	NoFix	Processor Can Enter a Livelock Condition under Certain Conditions When FP Exception Is Pending
W8			X	NoFix	Write Cycle of Write Combining Memory Type Does Not Self Snoop
W9			X	NoFix	Performance Monitoring Event that counts Floating Point Computational Exceptions (11h) Is Not Accurate
W10			X	NoFix	Inconsistent Reporting of Data Breakpoints on FP (Intel® MMX technology) loads
W11			X	NoFix	Code Breakpoint May Be Taken after POP SS Instruction If It Is followed by an Instruction That Faults
W12			X	NoFix	SysEnter and SysExit instructions May Write Incorrect Requestor Privilege Level (RPL) in the FP Code Segment Selector (FCS)

NO.	Steppings			Plans	ERRATA
	B-1 CPU Signature= 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature= 0695h		
W13	X	X	X	NoFix	Memory Aliasing with Inconsistent A and D Bits May Cause Processor Deadlock
W14	X	X	X	NoFix	RDMSR or WRMSR to Invalid MSR Address May Not Cause GP Fault
W15			X	NoFix	FP Tag Word Corruption
W16	X	X	X	NoFix	Unable to Disable Reads/Writes to Performance Monitoring Related MSRs
W17	X	X	X	NoFix	Move to Control Register Instruction May Generate a Breakpoint Report
W18			X	NoFix	REP MOVS Operation in Fast String Mode Continues in That Mode When Crossing Into a Page with a Different Memory Type
W19			X	NoFix	The FXSAVE, STOS, or MOVS Instruction May Cause a Store Ordering Violation When Data Crosses a Page with a UC Memory Type
W20			X	NoFix	Machine Check Exception May Occur Due to Improper Line Eviction in the IFU
W21			X	NoFix	POPF and POPFD Instructions That Set the Trap Flag Bit May Cause Unpredictable Processor Behavior
W22			X	NoFix	Performance Event Counter Returns Incorrect Value on L2_LINES_IN Event
W23			X	NoFix	VM Bit Will Be Cleared on a Double Fault Handler
W24	X	X	X	NoFix	Code Fetch Matching Disabled Debug Register May Cause Debug Exception
W25	X	X	X	NoFix	Upper Four PAT Entries Not Usable with Mode B or Mode C Paging
W26	X	X	X	NoFix	SSE/SSE2 Streaming Store Resulting in a Self-Modifying Code (SMC) Event May Cause Unexpected Behavior
W27	X	X		NoFix	Error in Instruction Fetch Unit (IFU) Can Result in an Erroneous Machine Check-Exception (#MC)
W28					Removed, see Erratum W3
W29					Removed, see Erratum W4
W30					Removed, see Erratum W5
W31	X	X	X	NoFix	Page with PAT (Page Attribute Table) Set to USWC (Uncacheable Speculative Write Combine) While Associated MTRR (Memory Type Range Register) is UC (Uncacheable) May Consolidate to UC
W32	X	X	X	NoFix	Under Certain Conditions LTR (Load Task Register) Instruction May Result in System Hang
W33	X	X	X	NoFix	Loading from Memory Type USWC (Uncacheable Speculative Write Combine) May Get Its Data Internally Forwarded from a Previous Pending Store

NO.	Steppings			Plans	ERRATA
	B-1 CPU Signature= 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature= 0695h		
W34	X	X	X	NoFix	FXSAVE after FNINIT Without an Intervening FP (Floating Point) Instruction May Save Uninitialized Values for FDP (x87 FPU Instruction Operand (Data) Pointer Offset) and FDS (x87 FPU Instruction Operand (Data) Pointer Selector)
W35	X	X	X	NoFix	FSTP (Floating Point Store) Instruction Under Certain Conditions May Result In Erroneously Setting a Valid Bit on an FP (Floating Point) Stack Register
W36		X		NoFix	An Execute Disable Bit Violation May Occur on a Data Page-Fault
W37		X		NoFix	CPUID Leaf 0x80000006 May Provide the Incorrect Value for an 8-Way Associative Cache
W38	X		X	PlanFix	Snoops during the Execution of a HLT (Halt) Instruction May Lead to Unexpected System Behavior
W39	X	X	X	NoFix	Invalid Entries in Page-Directory-Pointer-Table Register (PDPTR) May Cause General Protection (#GP) Exception if the Reserved Bits are Set to One
W40	X	X	X	NoFix	INIT Does Not Clear Global Entries in the TLB
W41	X	X	X	NoFix	Use of Memory Aliasing with Inconsistent Memory Type May Cause System Hang
W42	X	X	X	NoFix	Machine Check Exception May Occur When Interleaving Code between Different Memory Types

Number	SPECIFICATION CHANGES
	There are no Specification Changes in this Specification Update revision.

Number	SPECIFICATION CLARIFICATIONS
W1	Specification Clarification with Respect to Time-Stamp Counter
W2	Thermal Diode Offset Specification Clarification

Number	DOCUMENTATION CHANGES
	There are no Documentation Changes in this Specification Update revision.

§

Identification Information

The Celeron M processor can be identified by the following values:

Family[1]	Model[2]	Model[2] (Celeron M on 90 nm process)	Brand ID [3]
0110	1001	1101	00010010

T

NOTES:

1. The Family corresponds to bits [11:8] of the EDX register after Reset, bits [11:8] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register.
2. The Model corresponds to bits [7:4] of the EDX register after Reset, bits [7:4] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register.
3. The Brand ID corresponds to bits [7:0] of the EBX register after the CPUID instruction is executed with a1 in the EAX register.

Table 1. Intel® Celeron® M Identification

S-Spec	Processor number	Product Stepping	L2 Cache Size (bytes)	CPU Signature	Core Frequency	Bus Frequency	Voltage	Package Micro-FCBGA-Pb= Micro-FCBGA Lead Free
SL8LP	383	C-0	1 M	06D8h	1.00 GHz	400 MHz	0.876 V – 0.956 V[1]	Micro-FCBGA
SL8LV	383	C-0	1 M	06D8h	1.00 GHz	400 MHz	0.876 V – 0.956 V[1]	Micro-FCBGA-Pb
SL8BP	383	C-0	1 M	06D8h	1.00 GHz	400 MHz	0.940 V	Micro-FCBGA
SL8BN	383	C-0	1 M	06D8h	1.00 GHz	400 MHz	0.940 V	Micro-FCBGA-Pb
SL8LQ	373	C-0	512K	06D8h	1.00 GHz	400 MHz	0.876 V – 0.956 V[1]	Micro-FCBGA
SL8LW	373	C-0	512K	06D8h	1.00 GHz	400 MHz	0.876 V – 0.956 V[1]	Micro-FCBGA-Pb
SL8A4	373	C-0	512K	06D8h	1.00 GHz	400 MHz	0.940 V	Micro-FCBGA
SL89S	373	C-0	512K	06D8h	1.00 GHz	400 MHz	0.940 V	Micro-FCBGA-Pb
SL8MN	380	C-0	1 M	06D8h	1.60 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCPGA
SL8MG	380	C-0	1 M	06D8h	1.60 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA
SL8MU	380	C-0	1 M	06D8h	1.60 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA-Pb
SL8MM	370	C-0	1 M	06D8h	1.50 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCPGA
SL8MF	370	C-0	1 M	06D8h	1.50 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA
SL8MT	370	C-0	1 M	06D8h	1.50 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA-Pb
SL86J	370	C-0	1 M	06D8h	1.50 GHz	400 MHz	1.260 V	Micro-FCPGA
SL86P	370	C-0	1 M	06D8h	1.50 GHz	400 MHz	1.260 V	Micro-FCBGA
SL86D	370	C-0	1 M	06D8h	1.50 GHz	400 MHz	1.260 V	Micro-FCBGA-Pb
SL8ML	360J	C-0	1 M	06D8h	1.40 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCPGA
SL8ME	360J	C-0	1 M	06D8h	1.40 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA

S-Spec	Processor number	Product Stepping	L2 Cache Size (bytes)	CPU Signature	Core Frequency	Bus Frequency	Voltage	Package Micro-FCBGA-Pb= Micro-FCBGA Lead Free
SL86K	360J	C-0	1 M	06D8h	1.40 GHz	400 MHz	1.260 V	Micro-FCPGA
SL86Q	360J	C-0	1 M	06D8h	1.40 GHz	400 MHz	1.260 V	Micro-FCBGA
SL8MK	350J	C-0	1 M	06D8h	1.30 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCPGA
SL8MD	350J	C-0	1 M	06D8h	1.30 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA
SL86L	350J	C-0	1 M	06D8h	1.30 GHz	400 MHz	1.260 V	Micro-FCPGA
SL86R	350J	C-0	1 M	06D8h	1.30 GHz	400 MHz	1.260 V	Micro-FCBGA
SL7LS	360	B-1	1 M	06D6h	1.4 GHz	400 MHz	1.260 V	Micro-FCPGA
SL7LR	360	B-1	1 M	06D6h	1.4 GHz	400 MHz	1.260 V	Micro-FCBGA
SL7RA	350	B-1	1 M	06D6h	1.3 GHz	400 MHz	1.260 V	Micro-FCPGA
SL7R9	350	B-1	1 M	06D6h	1.3 GHz	400 MHz	1.260 V	Micro-FCBGA
SL6N7	320	B-1	512K	0695h	1.30 GHz	400 MHz	1.356 V	Micro-FCPGA
SL6NM	320	B-1	512K	0695h	1.30 GHz	400 MHz	1.356 V	Micro-FCBGA
SL6N6	330	B-1	512K	0695h	1.40 GHz	400 MHz	1.356 V	Micro-FCPGA
SL6NL	330	B-1	512K	0695h	1.40 GHz	400 MHz	1.356 V	Micro-FCBGA
SL7MT	340	B-1	512K	0695h	1.50 GHz	400 MHz	1.356 V	Micro-FCBGA
SL7ME	340	B-1	512K	0695h	1.50 GHz	400 MHz	1.356 V	Micro-FCPGA
SL79S	n/a	B-1	512K	0695h	1.20 GHz	400 MHz	1.356 V	Micro-FCPGA
SL79T	n/a	B-1	512K	0695h	1.20 GHz	400 MHz	1.356 V	Micro-FCBGA
SL7GE	n/a	B-1	512K	0695h	600 MHz	400 MHz	1.004 V	Micro-FCBGA
SL7DB	n/a	B-1	512K	0695h	800 MHz	400 MHz	1.004 V	Micro-FCBGA
SL7DH	n/a	B-1	512K	0695h	900 MHz	400 MHz	1.004 V	Micro-FCBGA
SL7F7	353	B-1	512 K	06D6h	900 MHz	400 MHz	0.940 V	Micro-FCBGA
SL7QX	353	B-1	512K	06D6h	900 MHz	400 MHz	0.940 V	Micro-FCBGA-Pb

1. These are optimized Voltage ID (VID) values. Individual processor VID values may be calibrated during manufacturing such that two devices at the same speed may have different VID settings.

Component Marking Information

Figure 1. The Intel® Celeron® M Processor (Micro-FCPGA/FCBGA) S-Spec Markings

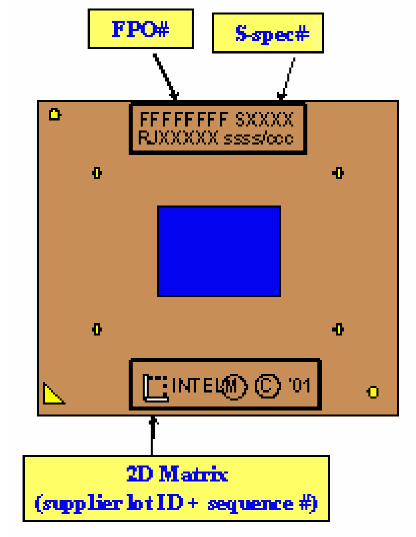


Figure 2. The Intel® Celeron® Processor ULV on 90 nm Process (Micro-FCBGA) S-Spec Markings

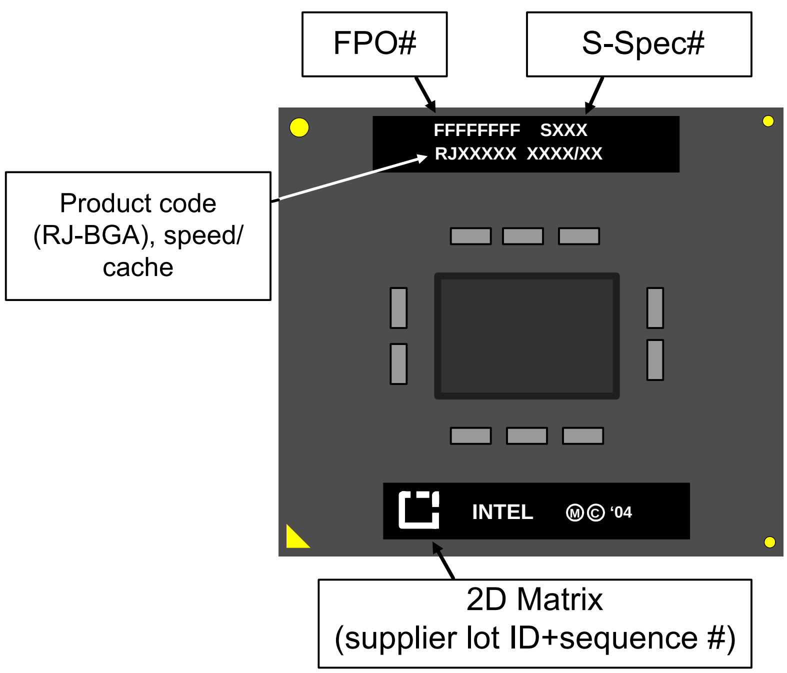
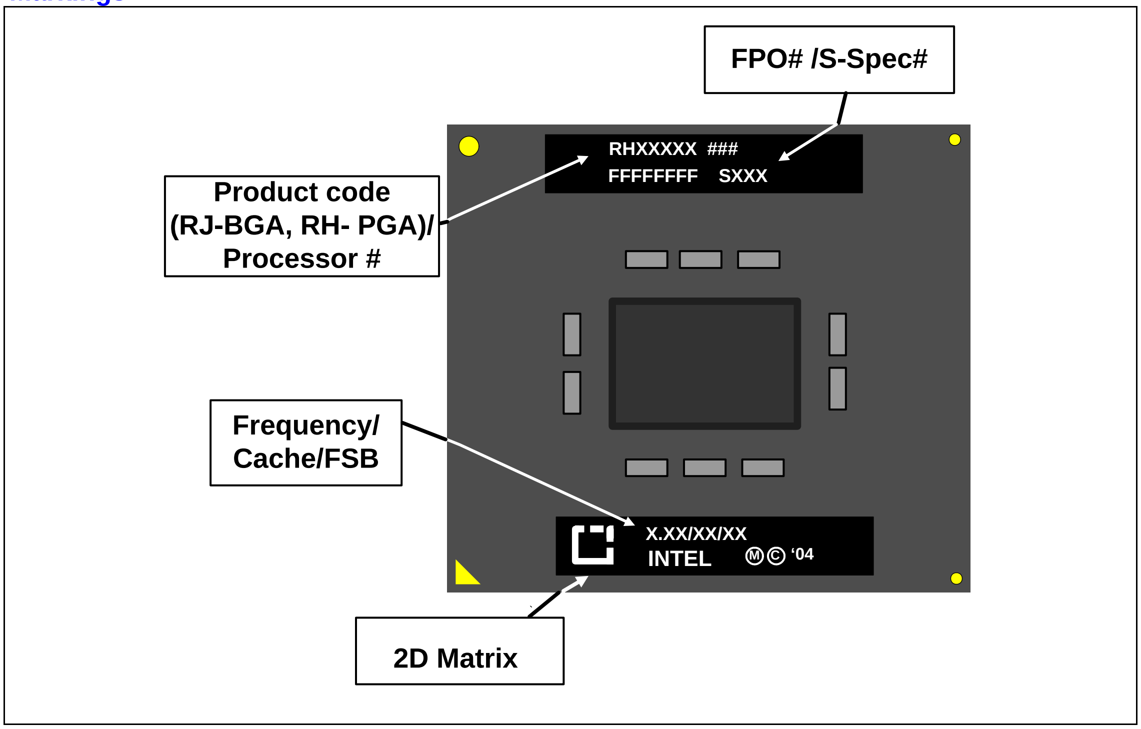


Figure 3.The Intel® Celeron® M Processor on 90 nm Process (Micro-FCPGA/FCBGA) S-Spec Markings



§

Errata

W1. Performance Monitoring Event that Counts the Number of Instructions Decoded (D0h) Is Not Accurate

Problem: The performance-monitoring event that counts the number of instructions decoded may have inaccurate results.

Implication: There is no functional impact of this erratum. However, the results/counts from this performance monitoring event should not be considered as being accurate

Workaround: None.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W2. RDTSC Instruction May Report the Wrong Time-Stamp Counter Value

Problem: The time-stamp counter is a 64-bit counter that is read in two 32-bit chunks. The counter incorrectly advances and therefore the two chunks may go out of synchronization causing the Read Time-stamp Counter (RDTSC) instruction to report the wrong time-stamp counter value

Implication: This erratum may cause software to see the wrong representation of processor time and may result in unpredictable software operation.

Workaround: It is possible for BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W3. Code Segment Limit Violation May Occur on 4-Gbyte Limit Check

Problem: Code Segment limit violation may occur on 4-Gbyte limit check when the code stream wraps around in a way that one instruction ends at the last byte of the segment and the next instruction begins at 0x0.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software, or system.

Workaround: Avoid code that wraps around segment limit.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W4. FST Instruction with Numeric and Null Segment Exceptions May Cause General Protection Faults to Be Missed and FP Linear Address (FLA) Mismatch

Problem: FST instruction combined with numeric and null segment exceptions may cause General Protection Faults to be missed and FP Linear Address (FLA) mismatch.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software, or system.

Workaround: None.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W5. Code Segment (CS) Is Wrong on SMM Handler When SMBASE Is Not Aligned

Problem: With SMBASE being relocated to a non-aligned address, during SMM entry the CS can be improperly updated which can lead to an incorrect SMM handler.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software, or system.

Workaround: Align SMBASE to 32-kB.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W6. IFU/BSU Deadlock May Cause System Hang

Problem: A lockable instruction with memory operand that spans across two pages may, given some rare internal conditions, hang the system.

Implication: When this erratum occurs, the system may hang. Intel has not observed this erratum with any commercially available software or system.

Workaround: Lockable data should always be contained in a single page.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W7. Processor Can Enter a Livelock Condition under Certain Conditions When FP Exception Is Pending

Problem: Processor clock modulation may be controlled via a processor register (IA32_THERM_CONTROL) or via the STPCLK# signal. While the processor clock is constantly being actively modulated at 12.5% and 25% duty cycles and there is a pending unmasked FP exception (ES pending), if you attempt a FP load (or Intel® MMX technology Mov instruction) and the load has an longer than typical latency the processor can enter a livelock.

Implication: When this erratum occurs, the processor will enter a livelock condition. Intel has not observed this erratum with any commercially available software or system.

Workaround: None.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W8. Write Cycle of Write Combining Memory Type Does Not Self Snoop

Problem: Write cycles of WC memory type do not self-snoop. This may result in data inconsistency- if the addresses of the WC data are aliased to WB memory type memory, which has been cached. In such a case, the internal caches will not be updated with the WC data sent on the system bus.

Implication: This condition may result in a data inconsistency. Intel has not observed this erratum with any commercially available software, system, nor components.

Workaround: Software should detect via the self-snoop bit in the CPUID features flags if the processor supports a self-snooping capability. Software should perform explicit memory management/flushing for aliased memory ranges on processor that do not self-snoop.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W9. Performance Monitoring Event That Counts Floating Point Computational Exceptions (11h) Is Not Accurate

Problem: Performance monitoring event that counts Floating Point Compare exceptions may have inaccurate results.

Implication: There is no functional impact of this erratum. However this Performance Monitoring Event should not be used when accurate performance monitoring is required.

Workaround: None.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W10. Inconsistent Reporting of Data Breakpoints on FP (Intel® MMX Technology) Loads

Problem: The reporting of data breakpoints on either FP or MMX technology loads is dependent upon the code faulting behavior prior to the execution of the load. If there is a fault pending prior to the execution of the load and FP exceptions are enabled there is a chance that data breakpoint on successive FP/MMX technology Loads may be reported twice.

Implication: Software debuggers should be aware of this possibility. There should be no implications to software operated outside of a debug environment.

Workaround: None.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W11. Code Breakpoint May Be Taken after POP SS Instruction If It Is followed by an Instruction That Faults

Problem: A POP SS instruction should inhibit all interrupts including Code Breakpoints until after execution of the following instruction. This allows sequential execution of POP SS and MOV eSP, eBP instructions without having an invalid stack during interrupt handling. However, a Code breakpoint may be taken after POP SS if it is followed by an instruction that faults, this result in a code breakpoint being reported on an unexpected instruction boundary since both instructions should be atomic.

Implication: This can result in a mismatched Stack Segment and SP. Intel has not observed this erratum with any commercially available software, or system.

Workaround: As recommended in the *IA32 Intel® Architecture Software Developer's Manual*, the use "POP SS" in conjunction with "MOV eSP, eBP" will avoid the failure since the "Mov" will not fault.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W12. SysEnter and SysExit Instructions May Write Incorrect Requestor Privilege Level (RPL) in the FP Code Segment Selector (FCS)

Problem: SysEnter and SysExit instructions may write incorrect RPL in the FP Code Segment selector (FCS). As a result of this, the RPL field in FCS may be corrupted.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software, or system.

Workaround: It is possible for BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W13. Memory Aliasing with Inconsistent A and D Bits May Cause Processor Deadlock

Problem: In the event that software implements memory aliasing by having two page directory entries (PDEs) point to a common page table entry (PTE) and the Accessed and Dirty bits for the two PDEs are allowed to become inconsistent, the processor may become deadlocked.

Implication: This erratum has not been observed with commercially available software.

Workaround: Software that needs to implement memory aliasing in this way should manage the consistency of the Accessed and Dirty bits.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W14. RDMSR or WRMSR to Invalid MSR Address May Not Cause GP Fault

Problem: The RDMSR and WRMSR instructions allow reading or writing of MSR's (Model Specific Registers) based on the index number placed in ECX. The processor should reject access to any reserved or unimplemented MSRs by generating #GP(0). However, there are some invalid MSR addressers for which the processor will not generate #GP(0). This erratum has not been observed with commercially available software.

Implication: For RDMSR, undefined values will be read into EDX:EAX. For WRMSR, undefined processor behavior may result.

Workaround: Do not use invalid MSR addresses with RDMSR or WRMSR.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W15. FP Tag Word Corruption

Problem: In some rare cases, fault information generated as the result of instruction execution may be incorrect. The result is an incorrect FP stack entry.

Implication: This erratum may result in corruption of the FP tag word in a way that a non-valid entry in the FP stack may become valid. The software is not expected to read a non-valid entry. If the software attempts to use the stack entry (which is expected to be empty) the result may be an erroneous "Stack overflow".

Workaround: Do not disable SSE/SSE2 in control register CR4 and avoid code segment limit violation.

Status: For the steppings affected, see the *Summary of Tables of Changes*.

W16. Unable to Disable Reads/Writes to Performance Monitoring Related MSRs

Problem: The Performance Monitoring Available bit in the miscellaneous processor features MSR (IA32_MISC_ENABLES.7) was defined so that when it is cleared to a 0, RDMSR/WRMSR/RDPMC instructions would return all zeros for reads of and prevent any writes to performance monitoring related MSRs. Currently it is possible to read from or write to performance monitoring related MSRs when the Performance Monitoring Available bit is cleared to a 0.

Implication: It is not possible to disallow reads and writes to the Performance Monitoring MSRs. Intel has not observed this erratum with any commercially available software or system.

Workaround: None.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W17. Move to Control Register Instruction May Generate a Breakpoint Report

Problem: A move (MOV) to control register (CR) instruction where control register is CR0, CR3 or CR4 may generate a breakpoint report.

Implication: MOV to control register instruction is not expected to generate a breakpoint report.

Workaround: Ignore breakpoint data from MOV to CR instruction.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W18. REP MOVS Operation in Fast String Mode Continues in That Mode When Crossing into a Page with a Different Memory Type

Problem: A fast "REP MOVS" operation will continue to be handled in fast mode when the string operation crosses a page boundary into an Uncacheable (UC) memory type. Also if the fast string operation crosses a page boundary into a WC memory region, the processor will not self-snoop the WC memory region. This may eventually result in incorrect data for the WC portion of the operation if those cache lines were previously cached as WB (through aliasing) and modified.

Implication: String elements should be handled by the processor at the native operand size in UC memory. In the event that the WB to WC aliasing case occurs, the end result could vary from normal software execution to potential software failure. Intel has not observed either aspects of this erratum in commercially available software.

Workaround: Software operating within Intel's recommendation will not require WB and WC memory aliased to the same physical address.

Status: For the steppings affected, see the *Summary Tables of Changes.*

W19. The FXSAVE, STOS, or MOVS Instruction May Cause a Store Ordering Violation When Data Crosses a Page with a UC Memory Type

Problem: If the data from an FXSAVE, STOS, or MOVS instruction crosses a page boundary from WB to UC memory type and this instruction is immediately followed by a second instruction that also issues a store to memory, the final data stores from both instructions may occur in the wrong order.

Implication: The impact of this store ordering behavior may vary from normal software execution to potential software failure. Intel has not observed this erratum in commercially available software.

Workaround: FXSAVE, STOS, or MOVS data must not cross page boundary from WB to UC memory type.

Status: For the steppings affected, see the *Summary Tables of Changes.*

W20. Machine Check Exception May Occur Due to Improper Line Eviction in the IFU

Problem: The processor is designed to signal an unrecoverable machine check exception (MCE) as a consistency checking mechanism. Under a complex set of circumstances involving multiple speculative branches and memory accesses there exists a one cycle long window in which the processor may signal a MCE in the instruction fetch unit (IFU) because instructions previously decoded have been evicted from the IFU. The one cycle long window is opened when an opportunistic fetch receives a partial hit on a previously executed but not as yet completed store resident in the store buffer. The resulting partial hit erroneously causes the eviction of a line from the IFU at a time when the processor is expecting the line to still be present. If the MCE for this particular IFU event is disabled, execution will continue normally.

Implication: While this erratum may occur on a system with any number of processors, the probability of occurrence increases with the number of processors. If this erratum does occur, a machine check exception will result. Note systems that implement an operating system that does not enable the Machine Check Architecture will be completely unaffected by this erratum (e.g., Windows* 95 and Windows*98).

Workaround: It is possible for BIOS code to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W21. POPF and POPFD Instructions That Set the Trap Flag Bit May Cause Unpredictable Processor Behavior

Problem: In some rare cases, POPF and POPFD instructions that set the Trap Flag (TF) bit in the EFLAGS register (causing the processor to enter Single-Step mode) may cause unpredictable processor behavior.

Implication: Single step operation is typically enabled during software debug activities, not during normal system operation.

Workaround: There is no workaround for single step operation in commercially available software. For debug activities on custom software the POPF and POPFD instructions could be immediately followed by a NOP instruction to facilitate correct execution.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W22. Performance Event Counter Returns Incorrect Value on L2_LINES_IN Event

Problem: The performance event counter returns an incorrect value on L2_LINES_IN event (EMON event #24H) when the L2 cache is disabled.

Implication: Due to this erratum, L2_LINES_IN performance event counter should not be monitored while the L2 cache is disabled. This erratum has no functional impact.

Workaround: Ignore L2_LINES_IN event when the L2 cache is disabled.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W23. VM Bit Will Be Cleared on a Double Fault Handler

Problem: Following a task switch to a Double Fault Handler that was initiated while the processor was in virtual-8086 (VM86) mode, the VM bit will be incorrectly cleared in EFLAGS.

Implication: When the OS recovers from the double fault handler, the processor will no longer be in VM86 mode

Workaround: None

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W24. Code Fetch Matching Disabled Debug Register May Cause Debug Exception

Problem: The bits L0-3 and G0-3 enable breakpoints local to a task and global to all tasks, respectively. If one of these bits is set, a breakpoint is enabled, corresponding to the addresses in the debug registers DR0-DR3. If at least one of these breakpoints is enabled, any of these registers are *disabled* (i.e., L*n* and G*n* are 0), and RW*n* for the disabled register is 00 (indicating a breakpoint on instruction execution), normally an instruction fetch will not cause an instruction-breakpoint fault based on a match with the address in the disabled register(s). However, if the address in a disabled register matches the address of a code fetch which also results in a page fault, an instruction-breakpoint fault will occur.

Implication: While debugging software, extraneous instruction-breakpoint faults may be encountered if breakpoint registers are not cleared when they are disabled. Debug software which does not implement a code breakpoint handler will fail, if this occurs. If a handler is present, the fault will be serviced. Mixing data and code may exacerbate this problem by allowing disabled data breakpoint registers to break on an instruction fetch.

Workaround: The debug handler should clear breakpoint registers before they become disabled.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W25. Upper Four PAT Entries Not Usable with Mode B or Mode C Paging

Problem: The page attribute table (PAT) contains eight entries, which must all be initialized and considered when setting up memory types for the Pentium III processor. However, in Mode B or Mode C paging, the upper four entries do not function correctly for 4-Kbyte pages. Specifically, bit 7 of page table entries that translate addresses to 4-Kbyte pages should be used as the upper bit of a 3-bit index to determine the PAT entry that specifies the memory type for the page. When Mode B (CR4.PSE = 1) and/or Mode C (CR4.PAE) are enabled, the processor forces this bit to zero when determining the memory type regardless of the value in the page table entry. The upper four entries of the PAT function correctly for 2-Mbyte and 4-Mbyte large pages (specified by bit 12 of the page directory entry for those translations).

Implication: Only the lower four PAT entries are useful for 4-kB translations when Mode B or C paging is used. In Mode A paging (4-Kbyte pages only), all eight entries may be used. All eight entries may be used for large pages in Mode B or C paging.

Workaround: None identified.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W26. SSE/SSE2 Streaming Store Resulting in a Self-Modifying Code (SMC) Event May Cause Unexpected Behavior

Problem: An SSE or SSE2 streaming store that results in a self-modifying code (SMC) event may cause unexpected behavior. The SMC event occurs on a full address match of code contained in L1 cache.

Implication: Due to this erratum, any of the following events may occur:

1. A data access break point may be incorrectly reported on the instruction pointer (IP) just before the store instruction.
2. A non-cacheable store can appear twice on the external bus (the first time it will write only 8 bytes, the second time it will write the entire 16 bytes).

Intel has not observed this erratum with any commercially available software. This erratum has been seen in a synthetic test environment.

Workaround: None.

Status: For the steppings affected, see the *Summary of Tables of Changes*

W27. Error in Instruction Fetch Unit (IFU) Can Result in an Erroneous Machine Check-Exception (#MC)

Problem: A rare combination of events including the generation of a bus lock(s), the execution of a WBINVD instruction, and a page accessed or dirty bit assist may result in an erroneous Machine Check-Exception (#MC).

Implication: Due to this erratum, unexpected machine check-exception (#MC) is generated. Intel has not been able to reproduce this erratum with commercially available software.

Workaround: It is possible for BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W28. Removed; See Erratum W3.

W29. Removed; See Erratum W4.

W30. Removed; See Erratum W5.

W31. Page with PAT (Page Attribute Table) Set to USWC (Uncacheable Speculative Write Combine) While Associated MTRR (Memory Type Range Register) Is UC (Uncacheable) May Consolidate to UC

Problem: A page whose PAT memory type is USWC while the relevant MTRR memory type is UC, the consolidated memory type may be treated as UC (rather than WC as specified in *IA-32 Intel® Architecture Software Developer's Manual*).

Implication: When this erratum occurs, the memory page may be as UC (rather than WC). This may have a negative performance impact.

Workaround: None identified.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W32. Under Certain Conditions LTR (Load Task Register) Instruction May Result in System Hang

Problem: A LTR instruction may result in a system hang if all the following conditions are met:

1. Invalid data selector of the TR (Task Register) resulting with either #GP (General Protection

Fault) or #NP (Segment Not Present Fault).

2. GDT (Global Descriptor Table) is not 8-bytes aligned.

3. Data BP (breakpoint) is set on cache line containing the descriptor data.

Implication: This erratum may result in system hang if all conditions have been met. This erratum has not been observed in commercial operating systems or software. For performance reasons, GDT is typically aligned to 8-bytes.

Workaround: Align GDT to 8-bytes.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W33. Loading from Memory Type USWC (Uncacheable Speculative Write Combine) May Get Its Data Internally Forwarded from a Previous Pending Store

Problem: A load from memory type USWC may get its data internally forwarded from a pending store. As a result, the expected load may never be issued to the external bus.

Implication: When this erratum occurs, a USWC load request may be satisfied without being observed on the external bus. There are no known usage models where this behavior results in any negative side-effects.

Workaround: Do not use memory type USWC for memory that has read side-effects.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W34. FXSAVE after FNINIT without an Intervening FP (Floating Point) Instruction May Save Uninitialized Values for FDP (x87 FPU Instruction Operand (Data) Pointer Offset) and FDS (x87 FPU Instruction Operand (Data) Pointer Selector)

Problem: An FXSAVE after FNINIT without an intervening FP instruction may save uninitialized values for FDP and FDS.

Implication: When this erratum occurs, the values for FDP/FDS in the FXSAVE structure may appear to be random values. These values will be initialized by the first FP instruction executed after the FXRSTOR that restored the saved floating point state. Any FP instruction with memory operand will initialize FDP/FDS. Intel has not observed this erratum with any commercially available software.

Workaround: After an FINIT, do not expect the FXSAVE memory image to be correct until at least one FP instruction with a memory operand has been executed.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W35. FSTP (Floating Point Store) Instruction Under Certain Conditions May Result in Erroneously Setting a Valid Bit on an FP (Floating Point) Stack Register

Problem: When an FSTP instruction with a PDE/PTE (Page Directory Entry/Page Table Entry) A/D bit update is followed by a user mode access fault due to a code fetch to a page that has supervisor only access permission, this may result in erroneously setting a valid bit of an FP stack register. The FP top of stack pointer is unchanged.

Implication: This erratum may cause an unexpected stack overflow.

Workaround: User mode code should not depend on being able to recover from illegal accesses to memory regions protected with supervisor only access when using FP instructions.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W36. An Execute Disable Bit Violation May Occur on a Data Page-Fault

Problem: Under a combination of internal events, unexpected Execute Disable violations may occur on data accesses that are Execute Disable protected.

Implication: This erratum may cause unexpected Execute Disable violations.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W37. CPUID Leaf 0x80000006 May Provide the Incorrect Value for an 8-Way Associative Cache

Problem: CPUID leaf 0x80000006 may return 0x8 in ECX [15:12] to indicate 8-way associative cache, but the correct encoding for an 8-way associative cache is 0x6.

Implication: Software that depends on the associativity of the cache may not function correctly.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W38. Snoops during the Execution of a HLT (Halt) Instruction May Lead to Unexpected System Behavior

Problem: If during the execution of a HLT instruction an external snoop causes an eviction from the instruction fetch unit (IFU) instruction cache, the processor may, on exit from the HLT state, erroneously read stale data from the victim cache.

Implication: This erratum may lead to unexpected system behavior. Intel has only observed this condition in non-mobile configurations.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W39. Invalid Entries in Page-Directory-Pointer-Table Register (PDPTR) May Cause General Protection (#GP) Exception If the Reserved Bits Are Set to One

Problem: Invalid entries in the Page-Directory-Pointer-Table Register (PDPTR) that have the reserved bits set to one may cause a General Protection (#GP) exception.

Implication: Intel has not observed this erratum with any commercially available software.

Workaround: Do not set the reserved bits to one when PDPTR entries are invalid.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W40. INIT Does Not Clear Global Entries in the TLB

Problem: INIT may not flush a TLB entry when:

1. The processor is in protected mode with paging enabled and the page global enable flag is set (PGE bit of CR4 register)
2. G bit for the page table entry is set
3. TLB entry is present in TLB when INIT occurs

Implication: Software may encounter unexpected page fault or incorrect address translation due to a TLB entry erroneously left in TLB after INIT.

Workaround: Write to CR3, CR4 or CR0 registers before writing to memory early in BIOS code to clear all the global entries from TLB.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W41. Use of Memory Aliasing with Inconsistent Memory Type May Cause System Hang

Problem: Software that implements memory aliasing by having more than one linear addresses mapped to the same physical page with different cache types may cause the system to hang. This would occur if one of the addresses is non-cacheable used in code segment and the other a cacheable address. If the cacheable address finds its way in instruction cache, and non-cacheable address is fetched in IFU, the processor may invalidate the non-cacheable address from the fetch unit. Any micro-architectural event that causes instruction restart will expect this instruction to still be in fetch unit and lack of it will cause system hang.

Implication: This erratum has not been observed with commercially available software.

Workaround: Although it is possible to have a single physical page mapped by two different linear addresses with different memory types, Intel has strongly discouraged this practice as it may lead to undefined results. Software that needs to implement memory aliasing should manage the memory type consistency.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

W42. Machine Check Exception May Occur When Interleaving Code between Different Memory Types

Problem: A small window of opportunity exists where code fetches interleaved between different memory types may cause a machine check exception. A complex set of micro-architectural boundary conditions is required to expose this window.

Implication: Interleaved instruction fetches between different memory types may result in a machine check exception. The system may hang if machine check exceptions are disabled. Intel has not observed the occurrence of this erratum while running commercially available applications or operating systems.

Workaround: Software can avoid this erratum by placing a serializing instruction between code fetches between different memory types.

Status: For the steppings affected, see the *Summary of Tables of Changes.*

§

Specification Changes

There are no Specification Changes in this Specification Update revision.

§

Specification Clarifications

All Specification Clarifications will be incorporated into a future version of the appropriate Mobile Pentium 4 processor documentation.

W1. Specification Clarification with Respect to Time-stamp Counter

In the “Debugging and Performance Monitoring” section (Sections 15.8, 15.10.9 and 15.10.9.3) of the *IA-32 Intel® Architecture Software Developer’s Manual Volume 3: System Programming Guide*, the Time-stamp Counter definition has been updated to include support for the future processors. This change will be incorporated in the next revision of the *IA-32 Intel® Architecture Software Developer’s Manual*.

15.8 Time-stamp Counter

The IA-32 architecture (beginning with the Pentium processor) defines a time-stamp counter mechanism that can be used to monitor and identify the relative time occurrence of processor events. The counter’s architecture includes the following components:

- **TSC flag —** A feature bit that indicates the availability of the time-stamp counter. The counter is available in an IA-32 processor implementation if the function CPUID.1:EDX.TSC[bit 4] = 1.
- **IA32_TIME_STAMP_COUNTER MSR** (called TSC MSR in P6 family and Pentium processors) **—** The MSR used as the counter.
- **RDTSC instruction —** An instruction used to read the time-stamp counter.
- **TSD flag —** A control register flag is used to enable or disable the time-stamp counter (enabled if CR4.TSD[bit 2] = 1).

The time-stamp counter (as implemented in the P6 family, Pentium, Pentium M, Pentium 4, and Intel Xeon processors) is a 64-bit counter that is set to 0 following a RESET of the processor. Following a RESET, the counter will increment even when the processor is halted by the HLT instruction or the external STPCLK# pin. Note that the assertion of the external DPSLP# pin may cause the time-stamp counter to stop.

Members of the processor families increment the time-stamp counter differently:

- For Pentium M processors (family [06H], models [09H, 0DH]); for Pentium 4 processors, Intel Xeon processors (family [0FH], models [00H, 01H, or 02H]); and for P6 family processors: the time-stamp counter increments with every internal processor clock cycle. The internal processor clock cycle is determined by the current core-clock to bus-clock ratio. Intel SpeedStep® technology transitions may also impact the processor clock.
- For Pentium 4 processors, Intel Xeon processors (family [0FH], models [03H and higher]): the time-stamp counter increments at a constant rate. That rate may be set by the maximum core-clock to bus-clock ratio of the processor or may be set by the frequency at which the processor is booted. The specific processor configuration determines the behavior. Constant TSC behavior ensures that the duration of each clock tick

is uniform and supports the use of the TSC as a wall clock timer even if the processor core changes frequency. This is the architectural behavior moving forward.

Note: To determine average processor clock frequency, Intel recommends the use of Performance Monitoring logic to count processor core clocks over the period of time for which the average is required. See Section 15.10.9 and Appendix A in this manual for more information.

The RDTSC instruction reads the time-stamp counter and is guaranteed to return a monotonically increasing unique value whenever executed, except for a 64-bit counter wraparound. Intel guarantees that the time-stamp counter will not wraparound within 10 years after being reset. The period for counter wrap is longer for Pentium 4, Intel Xeon, P6 family, and Pentium processors.

Normally, the RDTSC instruction can be executed by programs and procedures running at any privilege level and in virtual-8086 mode. The TSD flag allows use of this instruction to be restricted to programs and procedures running at privilege level 0. A secure operating system would set the TSD flag during system initialization to disable user access to the time-stamp counter. An operating system that disables user access to the time-stamp counter should emulate the instruction through a user-accessible programming interface.

The RDTSC instruction is not serializing or ordered with other instructions. It does not necessarily wait until all previous instructions have been executed before reading the counter. Similarly, subsequent instructions may begin execution before the RDTSC instruction operation is performed.

The RDMSR and WRMSR instructions read and write the time-stamp counter, treating the time-stamp counter as an ordinary MSR (address 10H). In the Pentium 4, Intel Xeon, and P6 family processors, all 64-bits of the time-stamp counter are read using RDMSR (just as with RDTSC). When WRMSR is used to write the time-stamp counter on processors before family [0FH], models [03H, 04H]: only the low order 32-bits of the time-stamp counter can be written (the high-order 32 bits are cleared to 0). For family [0FH], models [03H, 04H]: all 64 bits are writeable.

15.10.9 Counting Clocks

The count of cycles, also known as clockticks, forms the basis for measuring how long a program takes to execute. Clockticks are also used as part of efficiency ratios like cycles per instruction (CPI). Processor clocks may stop ticking under circumstances like the following:

- The processor is halted when there is nothing for the CPU to do. For example, the processor may halt to save power while the computer is servicing an I/O request. When Hyper-Threading Technology is enabled, both logical processors must be halted for performance-monitoring counters to be powered down.
- The processor is asleep as a result of being halted or because of a power-management scheme. There are different levels of sleep. In the some deep sleep levels, the time-stamp counter stops counting.

There are three ways to count processor clock cycles to monitor performance. These are:

- **Non-halted clockticks —** Measures clock cycles in which the specified logical processor is not halted and is not in any power-saving state. When Hyper-Threading Technology is enabled, these ticks can be measured on a per-logical-processor basis.
- **Non-sleep clockticks —** Measures clock cycles in which the specified physical processor is not in a sleep mode or in a power-saving state. These ticks cannot be measured on a logical-processor basis.
- **Time-stamp counter** — Some processor models permit clock cycles to be measured when the physical processor is not in deep sleep (by using the time-stamp counter and the RDTSC instruction). Note that such ticks cannot be measured on a per-logical-processor basis. See Section 10.8 for detail on processor capabilities.

The first two methods use performance counters and can be set up to cause an interrupt upon overflow (for sampling). They may also be useful where it is easier for a tool to read a performance counter than to use a time-stamp counter (the timestamp counter is accessed using the RDTSC instruction).

For applications with a significant amount of I/O, there are two ratios of interest:

- **Non-halted CPI —** Non-halted clockticks/instructions retired measures the CPI for phases where the CPU was being used. This ratio can be measured on a logical-processor basis when Hyper-Threading Technology is enabled.
- **Nominal CPI —** Time-stamp counter ticks/instructions retired measures the CPI over the duration of a program, including those periods when the machine halts while waiting for I/O.

15.10.9.3 Incrementing the Time-stamp Counter

The time-stamp counter increments when the clock signal on the system bus is active and when the sleep pin is not asserted. The counter value can be read with the RDTSC instruction.
The time-stamp counter and the non-sleep clockticks count may not agree in all cases and for all processors. See Section 10.8 for more information on counter operation.

W2. Thermal Diode Offset Specification Clarification

The following text has been added to *Intel® Celeron® M Processor on 90-nm Process Datasheet* chapter 5, section 5.1.2, *Thermal Diode Offset:*

The thermal diode offset model specific register, THERM_DIODE_OFFSET, is located at offset 03Fh accessed as a QWord. Bits 7:0 contain the thermal diode offset value in 0.5 °C resolution. This value should be subtracted from the diode measurement after the thermal diode ideality adjustments are made. Values from bits 7:0 of the MSR are interpreted as follows:

‘00000000’ = 0 °C
‘00000001’ = +0.5 °C
‘00000010’ = +1 °C
...
‘01111111’ = +63.5 °C
‘11111111’ = -0.5 °C
‘11111110’ = -1 °C
...
‘1000000’ = -64 °C

Appended to footnote of Table 5-7, *Thermal Diode Specification*:
Offset value is programmed in processor Model Specific Register 03Fh, “THERM_DIODE_OFFSET”

§

Documentation Changes

There are no Documentation Changes in this Specification Update revision.

§